IPCrypt: Optimal, Practical Encryption of IP Addresses for Privacy and Measurement

Frank Denis

Fastly Inc.

Abstract

This paper introduces practical methods for encrypting IPv4/IPv6 addresses while preserving utility in logs, telemetry, and third-party data exchange. We focus on three practical goals: (i) format-compatible encryption that keeps outputs in the IPv6 address space and handles IPv4 inputs canonically; (ii) prefix-preserving encryption that retains network structure for analytics while hiding host identity; and (iii) non-deterministic encryption that resists correlation while remaining compact and invertible. We present IPCrypt, a family of methods that meet these goals and rely only on standard primitives. We give deterministic, prefix-preserving, and two non-deterministic variants, with security models and arguments under standard assumptions, plus explicit usage bounds and operating limits.

We also relate each variant to known efficiency lower bounds (ciphertext expansion and primitive calls) and state our claims within deployable parameter ranges. The methods operate at block granularity and admit straightforward key management.

1 Introduction

IP addresses are routinely collected in logs, telemetry pipelines, and research datasets. These data enable abuse mitigation, capacity planning, and empirical research, but unprotected addresses pose significant privacy risks. Ad hoc truncation or hashing fails to offer sound confidentiality and frequently breaks operational workflows. We seek encryption mechanisms that support three prevalent use cases:

- **Format-compatible identifiers**: map addresses into the IPv6 textual space so existing dual-stack tooling keeps working, and note how to support legacy IPv4-only fields.
- **Prefix-preserving analytics**: maintain network prefix relations (e.g., /24, /64) to enable measurement and aggregation while hiding the concrete prefix and host bits.
- **Anti-correlation**: ensure the same address encrypts to different ciphertexts across events to suppress linkability; repeats occur only when the same tweak is reused (identical (X, T) yields the same ciphertext by design).

We present IPCrypt, a set of concrete methods that package conservatively chosen constructions with explicit domain modeling for IP addresses.

Design goals and constraints. Our deployments target: (i) *compactness* (16–32 bytes per record), (ii) *speed* (single-block or few-block processing), (iii) *interoperability* (platform-agnostic), and (iv) *sound security models* (PRP/PRF/TBC under standard assumptions). We normalize IPv4 and IPv6 to a single 16-byte representation and specify conversion in both directions. We assume systems already handle IPv6 representations, as is typical in

modern dual-stack infrastructure. If a deployment must keep the ciphertext of an IPv4 address syntactically IPv4, reserve IPv6-mapped storage (so deterministic ciphertexts are accepted) or add a 32-bit FPE layer; Section 3 discusses the implications.

Proposals.

- A deterministic 128-bit permutation using one AES call. It maps addresses compactly within the IPv6 presentation format and outlines shims for strict IPv4-only fields (Section 3).
- A prefix-preserving scheme that sets each bit using a PRF of the current prefix. We instantiate the PRF as the sum of two independently keyed AES permutations, preserving network structure (Section 4).
- Two non-deterministic, tweakable variants: KIASU-BC with 64-bit tweaks (24-byte output) and single-block XTS (XEX with encrypted tweak) with 128-bit tweaks (32-byte output). These achieve anti-correlation with explicit collision bounds at scale (Section 5).
- An efficiency comparison against known lower bounds on expansion and primitive calls, with the assumptions behind the trade-offs (Section 7).

On scope and claims. We do not claim new primitives. All security claims reduce to standard assumptions: AES as a PRP, established TBCs (KIASU-BC, XEX/XTS), and standard FPE guidance. We avoid brittle constant factors (e.g., raw Gbps). Instead, we report primitive-call counts and asymptotics so results are architecture-neutral and comparable across platforms.

2 Preliminaries and Domain Modeling

Addresses and representation. We treat an IP address as an element of $\{0,1\}^{128}$: an IPv6 address is taken in network byte order [HD06]; an IPv4 address $a \in \{0,1\}^{32}$ is encoded as an IPv4-mapped IPv6 address $00\ldots00\,\texttt{ff ff}\,a$ (10 zero bytes, 2 bytes `ff ff`, then the 4 IPv4 bytes). This encoding is injective and ensures a single 16-byte representation for both families. Conversion back to text form follows RFC 5952 for IPv6 rendering [KK10]. When the 16-byte value has the IPv4-mapped prefix, IPv4 is rendered; otherwise IPv6.

Primitives and notation. Let $E_K : \{0,1\}^{128} \to \{0,1\}^{128}$ denote AES-128 under key K [Nat01]. Our design relies on the usual PRP/PRF/TBC definitions; tweaks are public strings, and random tweaks hide cross-record linkage except for identical (X, T) pairs, which inevitably match.

2.1 Threat Model and Goals

Adversaries see ciphertexts, may issue chosen-plaintext queries, and obtain standard side information (counts, timestamps). We target only confidentiality under operational constraints:

- **Deterministic variant**: PRP security over the 128-bit domain; adversary learns equality of identical inputs by design.
- **Prefix-preserving variant**: Security against distinguishing from an ideal prefix-preserving random bijection defined via a truly random function over prefixes (Section 4.1).

- **Non-deterministic variants**: IND-CPA security in the TBC sense for chosen tweaks; random tweaks remove cross-record linkability except on exact (input, tweak) repeats.

We assume constant-time primitives to prevent key recovery through timing channels, sound key storage, and canonical address parsing/rendering (RFC 4291, RFC 5952). Branches on public values (e.g., IPv4 vs IPv6 distinction) do not compromise security.

3 IPCrypt-Deterministic: Deterministic Format-Compatible Encryption

We define a bijection $F_K : \{0,1\}^{128} \to \{0,1\}^{128}$ by $F_K(x) = E_K(x)$ where $K \in \{0,1\}^{128}$. The encryption of an address A is $\mathsf{Enc}_K(A) = \text{toIP}(F_K(\text{to16}(A)))$ where to16 and toIP are the conversions in Section 2.

Algorithm 1 $\mathsf{EncDet}(K, A)$ and $\mathsf{DecDet}(K, \hat{A})$

$x \leftarrow \text{to16}(A)$
$y \leftarrow E_K(x)$
return $\text{toIP}(y)$

$y \leftarrow \text{to16}(\hat{A})$
$x \leftarrow E_K^{-1}(y)$
return $\text{toIP}(x)$

Security. If AES is a secure 128-bit PRP, then F_K is a secure PRP on 128-bit strings; PRP indistinguishability directly lifts to the address domain through the encoding. Deterministic encryption is not IND-CPA for low min-entropy sources; this variant is intended when linkability of identical addresses is acceptable or desired (e.g., deduplication, rate limiting). This behavior aligns with the general caveat for deterministic encryption of identifiers.

IPv4 compatibility. Because E_K permutes the full 128-bit space, an IPv4 input encoded via `::ffff/96` will almost always map outside that subspace. EncDet therefore renders it as an IPv6 address. Deployments that must output dotted-quad strings should either treat ciphertext fields as IPv6 (which still preserves round-trip recovery) or add a 32-bit FPE layer around EncDet for IPv4-only sinks.

Efficiency discussion. Any bijection on 128 bits that attains PRP security with black-box access to a block cipher requires at least one primitive call per encryption. Our construction meets that call-count lower bound and has zero expansion beyond the 16-byte domain size, but this guarantee applies to the IPv6 presentation domain; installations that insist on IPv4-format outputs must budget the cost of the additional FPE layer noted above.

4 IPCrypt-PFX: Prefix-Preserving Encryption

Given two addresses that share an ℓ-bit prefix, a *prefix-preserving* encryption must output ciphertexts sharing the same ℓ-bit prefix. Let $P \in \{0,1\}^{\leq 128}$ be a prefix of the input address $X \in \{0,1\}^{128}$, ordered from MSB to LSB. We define a PRF $G_{K_1,K_2}(P) = E_{K_1}(\text{pad}(P)) \oplus E_{K_2}(\text{pad}(P))$, where $K_1, K_2 \in \{0,1\}^{128}$ are independent and $\text{pad}(\cdot)$ maps a prefix to a 128-bit block as

$$\text{pad}(P) = 0^{128-|P|-1}\, 1\, P\,,$$

placing a 1-bit delimiter immediately before P. For IPv4 values, processing starts at bit 96 (the IPv4-mapped boundary), but the top bits are still included in the prefix, preserving domain separation between IPv4 and IPv6.

Bit-wise encryption. Encryption proceeds from MSB to LSB. At step i with processed prefix P_i, compute $b_i = \mathrm{LSB}(G_{K_1,K_2}(P_i))$ and set $Y[i] = X[i] \oplus b_i$. The next prefix is $P_{i+1} = P_i \parallel X[i]$. Decryption recomputes b_i from the recovered prefix and inverts the XOR. Section A gives full algorithms.

Correctness and prefix preservation. By construction P_i is identical for any two inputs that share the first i bits; therefore the corresponding b_i are equal and the first i output bits match. The mapping is invertible since each step is a one-bit XOR with a prefix-determined mask, reconstructible during decryption.

4.1 Idealized Security Notion

Let $\mathcal{R}$ be the family of prefix-preserving bijections realized by choosing an independent random bit $r_P \in \{0,1\}$ for each possible prefix P (including the empty prefix and the IPv4 boundary prefix) and computing $Y[i] = X[i] \oplus r_{P_i}$ with $P_{i+1} = P_i \parallel X[i]$. We give the adversary oracle access to an unknown F that is either our construction with G instantiated by a PRF (real world) or a uniformly sampled $R \leftarrow \mathcal{R}$ (ideal world). The advantage is $\mathrm{Adv}^{\mathsf{pp-prp}}_{\mathcal{A}} = |\Pr[\mathcal{A}^F = 1] - \Pr[\mathcal{A}^R = 1]|$.

Security (keyed sum). Classic results show that the sum (XOR) of two independent n-bit pseudorandom permutations is a strong PRF with *n-bit* security [Luc00, Pat08, Pat09, NPV10, Pat17]. In our setting $G_{K_1,K_2}(\cdot) = E_{K_1}(\cdot) \oplus E_{K_2}(\cdot)$ is evaluated under secret, independent AES keys, so this keyed-PRP-to-PRF theorem applies directly. Replacing G by a truly random function over padded prefixes makes the scheme a sample from $\mathcal{R}$ by construction, and a standard PRF-hybrid reduces distinguishing our scheme to distinguishing G from random.

Numerics for n=128. Specializing the above, our keyed construction uses n=128 and thus inherits *128-bit PRF security* for G, i.e., distinguishing advantage grows on the order of $q/2^{128}$ (plus the PRP advantages for the two AES instances) [Luc00, Pat08, Pat09].

Complexity. For IPv4 (32 bits) the construction performs 2×32 AES calls; for IPv6 (128 bits) it performs 2×128 calls. During encryption, the encoder knows the full prefix sequence and can precompute mask bits in parallel: feed each padded prefix to the PRF and take one bit. Decryption is sequential across bits because each next prefix depends on the already decrypted bits; however, the two AES calls per bit are independent and can be executed in parallel. Implementations can cache $G(P)$ across addresses that share prefixes to amortize costs in analytics settings.

Key requirement. $K_1 \neq K_2$; if $K_1 = K_2$ then $G \equiv 0$ and the transformation reduces to the identity.

Efficiency discussion. Any prefix-preserving, key-dependent, invertible mapping that is computationally unpredictable must, in the black-box PRF model, consume at least one PRF evaluation per revealed bit of prefix; otherwise, the next-bit distribution conditioned on the prefix would be independent of a key-dependent value, enabling prediction. This yields a

per-address worst case of 32 evaluations for IPv4 and 128 for IPv6. In practice these are *upper bounds*: common prefixes can be precomputed and cached so repeats amortize close to zero additional work, and implementations can parallelize AES calls to reduce latency.

5 IPCrypt-ND and IPCrypt-NDX: Non-Deterministic Encryption

To prevent correlation, we use tweakable block ciphers with random, per-encryption tweaks. Ciphertexts are the concatenation of the tweak and the 16-byte encrypted address. We provide two instantiations with different tweak sizes. We focus on random tweaks over counters or permutation-based indices because they are stateless and easiest to deploy across distributed writers; counters/permutation require global coordination and risk reuse after crashes or replays. The residual birthday term is negligible with routine key rotation (Section 5.3).

5.1 IPCrypt-ND: KIASU-BC with 64-bit tweaks

Let $T \in \{0,1\}^{64}$ be uniformly random. Pad T to 128 bits by placing each 16-bit chunk at the head of each 32-bit lane (the remaining bytes are zero). The 128-bit key is expanded as in AES, and encryption follows the 10-round AES structure with the padded tweak XORed with the round key in every round (including round 0 and the final round). We define C as $E^{\text{KIASU-BC}}_{K,T}(X)$ and output $T \parallel C$ (24 bytes).

Choice of KIASU-BC. Among AES-based tweakable block ciphers, KIASU-BC offers strong performance and is simple to implement: it requires only XORing the tweak each round, so it drops into any existing AES implementation. Other standard TBCs (e.g., SKINNY/Deoxys) could be substituted without changing the interface or bounds [BJK+16, JNPS21]. This simplicity and speed make it a pragmatic choice for high-throughput deployments.

Bounds. The birthday bound for 64-bit tweaks gives an expected first tweak collision around $\approx 2^{32}$ encryptions under one key. An (X, T) collision reveals repetition but not X. Keys can be rotated well before this limit.

5.2 IPCrypt-NDX: single-block XTS with 128-bit tweaks

Let $T \in \{0,1\}^{128}$ be uniformly random and let the 32-byte key be $K = K_1 \parallel K_2$. We define ET as $E_{K_2}(T)$ and $C = E_{K_1}(X \oplus \text{ET}) \oplus \text{ET}$ (XEX with encrypted tweak [Rog04], the core of XTS [IEE08, Nat10]). The output is $T \parallel C$ (32 bytes). Collisions in T occur near $\approx 2^{64}$ encryptions per key.

Security and reuse. For both variants, chosen-plaintext security reduces to TBC security for a fixed tweak and to PRP security of AES for each tweak; random tweaks provide domain separation across events. Reusing the same tweak on *different* inputs is allowed under TBC definitions and does not, by itself, enable efficient distinguishing attacks. Equality of (X, T) across encryptions remains inherently visible.[1] Persistently reusing a single tweak T across many records reintroduces linkage within that T-bucket and undermines anti-correlation goals for that subset; see Section 5.3. Assuming KIASU-BC with 64-bit tweaks is a secure TBC over $\{0,1\}^{128}$, IPCrypt-nd inherits IND-CPA security in the tweakable sense, and with uniform random tweaks ciphertexts stay unlinkable except on exact (X, T) repeats. If XEX

[1] As usual, reusing tweaks with the same input reveals repetition; this is unavoidable for any invertible construction.

with encrypted tweak instantiated with AES (XTS core) acts as a PRP for each fixed tweak and K_1, K_2 are independent, the same reasoning gives IND-CPA security for IPCrypt-ndx; random 128-bit tweaks yield collision probability about $q^2/2^{129}$ over q encryptions per key.

Efficiency discussion. For a target of Q encryptions per key with negligible probability of a random tweak collision, information-theoretic bounds require $|T| \geq \log_2\left(\frac{Q(Q-1)}{2\varepsilon}\right)$ bits. Thus, achieving $Q \approx 2^{32}$ with small ε requires ≈ 64 bits; achieving $Q \approx 2^{64}$ requires ≈ 128 bits. Our 64- and 128-bit tweaks match those back-of-the-envelope requirements, but operators must still account for the practical meaning: a 64-bit tweak collides around 2^{32} encryptions (roughly a few seconds of full-line-rate flow logs in high-volume telemetry), so key rotation or per-stream tweak derivation is advised in such environments. See Section 5.3 for operational guidance on how to size and manage tweaks.

5.3 Operational Guidance on Tweak Repetition

What repetition reveals. With public per-encryption tweaks T, the ciphertext is $T \parallel C$ (Section 5). If the *same* pair (X, T) is used twice under one key, the same C reappears; the observer learns only that the exact (X, T) occurred again. Reusing the *same* T across *different* inputs creates a fixed-T PRP instance: duplicates of X within that T-bucket map to duplicate C values, enabling linkage within the bucket. Across *different* tweaks, the same X is unlinkable. Even under a fixed T, no address structure (prefixes, XORs, distances) is exposed.

Accidental collisions vs. systematic reuse. Tweaks are assumed uniformly random. For t-bit tweaks and q encryptions per key, the any-collision probability is about $q(q-1)/2^{t+1}$. Such random collisions may repeat the T label and, rarely, the (X, T) pair; they do not reveal X. In contrast, *systematic reuse* of one T for many records would enable deterministic linkage within that bucket (though not plaintext recovery) and should be eliminated.

Volume guidance. Tweak size is fixed by the variant: t=64 for IPCrypt-nd and t=128 for IPCrypt-ndx. For q encryptions per key, the any-collision probability is about $q(q-1)/2^{t+1}$. With IPCrypt-nd (t=64), Q=10^8 gives collision probability $\approx 2.7 \times 10^{-4}$, and the first collision is expected around $\approx 2^{32}$ encryptions. With IPCrypt-ndx (t=128), collisions remain negligible for typical telemetry scales (e.g., $Q \leq 10^{12}$).

Adversary's view. Without the key, an adversary cannot recover addresses or test guesses. They can: (i) detect exact repeats of (X, T), and (ii) group records that share a repeated T. Under our assumption of uniformly random tweaks, these groupings carry no semantics.

6 Domain Representation and Key Management

Address encoding. We provide unambiguous conversion algorithms between IP addresses and their 16-byte representation. For IPv4, we use the IPv4-mapped prefix; for decoding, we render IPv4 when that prefix appears. This encoding preserves the IPv4/IPv6 domain separation required for the prefix-preserving variant.

Algorithm 2 to16($\cdot$): address to 16 bytes

Parse A as IPv4 or IPv6.
if A is IPv4 **then**
    return 00^{80} `ff ff` A
else
    return IPv6 bytes in network order.
end if

Algorithm 3 toIP($\cdot$): 16 bytes to address

if first 10 bytes are 0 and next two are `ff ff` **then**
    Render last 4 bytes as IPv4.
else
    Render as IPv6 (RFC 5952).
end if

Key separation. To prevent cross-mode correlations, we derive per-variant subkeys from a single master key K_{m} using HKDF (RFC 5869 [KE10]) with distinct domain strings:

$$K_{\mathsf{det}} := \mathrm{HKDF}(K_{\mathsf{m}}, \text{“ipcrypt-det”})$$
$$K_{\mathsf{pfx}} := \mathrm{HKDF}(K_{\mathsf{m}}, \text{“ipcrypt-pfx”})$$
$$K_{\mathsf{nd}} := \mathrm{HKDF}(K_{\mathsf{m}}, \text{“ipcrypt-nd”})$$
$$K_{\mathsf{ndx}} := \mathrm{HKDF}(K_{\mathsf{m}}, \text{“ipcrypt-ndx”})$$

This ensures that using the same master key across different variants does not enable cross-variant attacks. The master key must be uniformly random (RFC 4086 [ESC05]). HKDF collisions between the derived 128-bit strings are astronomically unlikely, but deployments should still enforce $K_1 \neq K_2$ for the prefix-preserving transform during provisioning and re-derive in the (rare) event of equality to avoid the degenerate identity mapping noted in Section 4.

Non-deterministic output format. The non-deterministic variants produce ciphertexts as tweak $\|$ encrypted-block (24 bytes for IPCrypt-nd with 64-bit tweaks, 32 bytes for IPCrypt-ndx with 128-bit tweaks). The tweak must be transmitted or stored alongside the ciphertext to enable decryption.

7 Efficiency Relative to Lower Bounds

We compare each construction with simple lower bounds on two resources: (i) ciphertext expansion, and (ii) number of primitive calls (AES or TBC invocations).

Format-compatible (deterministic). Any keyed, invertible mapping on an n-bit domain with PRP security must evaluate at least one n-bit permutation on the input; otherwise, an adversary can distinguish by querying the identity or via simple linearity checks. Our scheme uses one AES call and no expansion, meeting that floor for the IPv6 presentation domain discussed in Section 3; strictly IPv4-formatted outputs necessarily incur the overhead of the additional FPE layer mentioned earlier.

Prefix-preserving. For a secure, key-dependent, prefix-preserving bijection on n bits, the i-th output bit must depend on the i-bit input prefix and on a key-dependent secret. Otherwise the next bit is predictable given the prefix. In the black-box model, at least one PRF evaluation per bit is required; thus at least n evaluations. IPCrypt uses one such evaluation per bit, instantiated as the sum of two AES permutations to obtain the concrete security bound of Section 4.1.

Anti-correlation (tweaked). If tweaks are random strings of length t, the expected first collision occurs around $\Theta(2^{t/2})$ encryptions. Therefore, for target volume Q with negligible collision probability ε, it is required that $t \gtrsim 2\log_2 Q + \log_2(1/\varepsilon)$ bits. Consequently the ciphertext must expand by at least t bits to carry the tweak. IPCrypt-nd with t=64 and IPCrypt-ndx with t=128 match these rough requirements for typical telemetry volumes and billion-scale deployments, respectively, subject to the collision-management guidance in Section 5.3.

8 Performance and Practicalities

All variants operate on a single 16-byte block per encryption, except the prefix-preserving scheme which performs two AES calls per bit. We report primitive-call counts (1 for det, $2n$ for pfx with $n \in \{32, 128\}$, 1 for nd/ndx) and exact ciphertext sizes (16, 4 or 16, 24, and 32 bytes, respectively). Non-deterministic variants add exactly the tweak length to the 16-byte ciphertext. Because outputs remain compact, they integrate cleanly with logging and telemetry infrastructures. We implemented IPCrypt in 10+ programming languages and found the methods straightforward to implement and deploy; reference implementations and bindings are available at `https://ipcrypt-std.github.io/`.

Why not generic FPE? NIST SP 800-38G FPE constructions (FF1/FF3) operate over general radix domains and use multi-round Feistel structures [Nat16, BRRS09, BR02]. FAST [DHV21] is a more recent alternative to 38G-style FPE. On our fixed 128-bit domain, the deterministic variant meets the single-call lower bound from Section 7 (one permutation call, zero expansion), whereas generic FPE needs multiple permutation calls and optional tweak or length overhead. As for FAST, we are not aware of production-grade implementations as of 2025, and reported performance remains substantially slower than commodity AES, making it ill-suited to our single-block address setting. For preserving network structure, generic FPE does not apply; prefix-preservation is distinct from format-preservation.

9 Related Work

Our work builds on several established cryptographic foundations. Format-preserving encryption has been extensively studied for structured domains [BR02, BRRS09, DHV21], though we achieve better efficiency for our fixed 128-bit domain by using direct permutation rather than generic FPE constructions. For prefix preservation, the networking community has long used Crypto-PAn-style designs [FXAM02, pro] that apply a PRF to each prefix bit. We adopt this proven template but strengthen the PRF instantiation: rather than a single permutation, we use the sum of two independent AES permutations.

Our non-deterministic variants employ tweakable block ciphers [LRW02]. We chose KIASU-BC [JNP14a, JNP14b] for its simplicity, requiring only XOR operations at each round, and the well-established XTS mode [Rog04, IEE08, Nat10], widely deployed in disk

encryption. Both provide the tweakability we need while remaining straightforward to implement atop existing AES code.

Crypto-PAn vs. IPCrypt-pfx. The classic Crypto-PAn construction [FXAM02] popularized prefix-preserving anonymization, but practical notes from deployments (e.g., Lucent's extensions[2]) surfaced issues that our IPCrypt-pfx instantiation addresses. First, Crypto-PAn's PRF input can repeat across successive bit positions. This may yield visible runs of identical mask bits (long sequences of 0s/1s in the XOR with the plaintext) and biased diffusion. Engineers mitigated this with ad-hoc pad updates or by sampling different output bit positions per round. In IPCrypt-pfx, the PRF input is a delimited, growing prefix block $(0^* 1\, P)$, which guarantees a fresh input at every step and eliminates repeated-round inputs. Second, we remove byte-order pitfalls seen in practice: we fix network byte order, define explicit MSB-to-LSB indexing, and pin the IPv4 start at the mapped boundary to ensure cross-platform reproducibility. Third, IPCrypt-pfx is IPv6-native and preserves IPv4/IPv6 domain separation (prefix processing starts at bit 96 for IPv4), whereas Crypto-PAn was originally specified for IPv4 and extended variably. Finally, our PRF affords a stronger multi-query security margin than a single-permutation instantiation, useful for billion-scale telemetry while keeping the same bitwise prefix-preserving interface.

10 Limitations and Misuse Considerations

Deterministic encryption reveals equality and supports cross-table joins; should be used only when this is acceptable. Prefix-preserving encryption intentionally reveals network structure and may enable coarse-grained inferences; it is unsuited for settings requiring full unlinkability. Non-deterministic variants provide no authenticity; they do not prevent substitution attacks if a decryption oracle is exposed. All variants protect only the address field; surrounding metadata (timestamps, ports, user agents) can still enable re-identification unless they are handled appropriately (cf. RFC 6973 [CT+13]).

11 Conclusion

We present IPCrypt, a set of deployable methods for encrypting IP addresses that meet three operationally critical goals. The constructions use only standard primitives, come with rigorous domain modeling, and align with the efficiency targets discussed throughout the paper. These properties make them suitable building blocks for privacy-preserving telemetry, measurement, and data sharing.

A Prefix-Preserving Encryption Pseudocode

For completeness, we provide explicit pseudocode for the prefix-preserving transform. For IPv4 inputs, processing starts at bit 96 so that only native 32 bits are transformed; for IPv6, at bit 0.

[2]See `https://sk.tl/8JkXJm5v`.

Algorithm 4 $\mathsf{EncPfx}(K_1, K_2, A)$

$X \leftarrow \text{to16}(A)$, $Y \leftarrow 0^{128}$
$i_0 \leftarrow 96$ if X is IPv4-mapped else 0; initialize pad as $0^{127}\,1$ or $0^{31}\,1\,0^{80}$ `ff ff` accordingly.
for $i = i_0$ to 127 **do**
    $e_1 \leftarrow E_{K_1}(\text{pad})$, $e_2 \leftarrow E_{K_2}(\text{pad})$, $b \leftarrow \text{LSB}(e_1 \oplus e_2)$
    $j \leftarrow 127 - i$, $Y[j] \leftarrow X[j] \oplus b$
    Left-shift pad by one and set its LSB to $X[j]$.
end for
return $\text{toIP}(Y)$

Decryption recomputes the same mask bits b from the already recovered prefix and inverts the XOR at each step.

References

[BJK+16] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich, and S. M. Sim. The skinny family of block ciphers and its low-latency variant mantis. In *CRYPTO 2016*, LNCS 9815, pages 123–153, 2016.

[BR02] J. Black and P. Rogaway. Format-preserving encryption. In *CT-RSA 2002*, LNCS 2271, pages 114–130, 2002.

[BRRS09] M. Bellare, T. Ristenpart, P. Rogaway, and T. Stegers. Format-preserving encryption. In *SAC 2009*, LNCS 5867, pages 295–312. Springer, 2009.

[CT+13] A. Cooper, H. Tschofenig, et al. Privacy considerations for internet protocols. RFC 6973, July 2013.

[DHV21] F. B. Durak, H. Horst, and S. Vaudenay. Fast: Format-preserving encryption via shortened aes tweakable block cipher. IACR Cryptology ePrint Archive 2021/1171, 2021.

[ESC05] D. Eastlake, J. Schiller, and S. Crocker. Randomness requirements for security. RFC 4086, June 2005.

[FXAM02] J. Fan, J. Xu, M. H. Ammar, and S. B. Moon. Prefix-preserving ip address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. In *10th IEEE International Conference on Network Protocols*, pages 280–289. IEEE, 2002.

[HD06] R. Hinden and S. Deering. IPv6 addressing architecture. RFC 4291, February 2006.

[IEE08] IEEE Computer Society. IEEE standard for cryptographic protection of data on block-oriented storage devices. IEEE Std 1619-2007, March 2008.

[JNP14a] J. Jean, I. Nikolić, and T. Peyrin. Kiasu v1.1. First-round submission to CAESAR competition, 2014.

[JNP14b] J. Jean, I. Nikolić, and T. Peyrin. Tweaks and keys for block ciphers: The TWEAKEY framework. In *ASIACRYPT 2014*, LNCS 8874, pages 274–288, 2014.

[JNPS21] J. Jean, I. Nikolić, T. Peyrin, and Y. Seurin. The deoxys aead family. *Journal of Cryptology*, 34(31), 2021.

[KE10] H. Krawczyk and P. Eronen. Hmac-based extract-and-expand key derivation function (hkdf). RFC 5869, May 2010.

[KK10] S. Kawamura and M. Kawashima. A recommendation for IPv6 address text representation. RFC 5952, August 2010.

[LRW02] M. Liskov, R. L. Rivest, and D. Wagner. Tweakable block ciphers. In *CRYPTO 2002*, LNCS 2442, pages 31–46, 2002.

[Luc00] S. Lucks. The sum of prps is a secure prf. In *EUROCRYPT 2000*, LNCS 1807, pages 470–484, 2000.

[Nat01] National Institute of Standards and Technology. Advanced encryption standard (aes). FIPS PUB 197, November 2001.

[Nat10] National Institute of Standards and Technology. Recommendation for block cipher modes of operation: The xts-aes mode for confidentiality on storage devices. NIST SP 800-38E, January 2010.

[Nat16] National Institute of Standards and Technology. Recommendation for block cipher modes of operation: Methods for format-preserving encryption. Technical Report SP 800-38G, NIST, March 2016.

[NPV10] V. Nachef, J. Patarin, and E. Volte. Introduction to mirror theory: Analysis of systems of linear equalities and linear non equalities for cryptography. IACR Cryptology ePrint Archive 2010/287, 2010.

[Pat08] J. Patarin. A proof of security in $o(2^n)$ for the xor of two random permutations. In *ICITS 2008*, LNCS 5155, pages 232–248. Springer, 2008.

[Pat09] J. Patarin. The "coefficients h" technique. In *SAC 2008*, LNCS 5381, pages 328–345. Springer, 2009.

[Pat17] J. Patarin. Mirror theory and cryptography. *Applicable Algebra in Engineering, Communication and Computing*, 28(4):321–338, 2017.

[pro] U. A. project. cryptopant: Ip address anonymization library. Software library.

[Rog04] P. Rogaway. Efficient instantiations of tweakable blockciphers and refinements to modes ocb and pmac. In *ASIACRYPT 2004*, LNCS 3329, pages 16–31. Springer, 2004.